Jani Enroth, 424851
Tietotekniikka
Vuosikurssi 2013
23.4.2019

Projektityön dokumentti – Älykäs reseptikirja

Yleiskuvaus

Älykäs reseptikirja on ohjelma, joka ylläpitää ruokareseptikirjaa, jonka avulla voi helposti etsiä kriteerit täyttävä ruokalaji. Käyttäjällä on ajanmukaiset tiedot ruoka-aineista, jotka ovat sillä hetkellä varastossa. Ohjelma voi etsiä ruokalajeja, jotka voidaan valmistaa käymättä kaupassa, tai ruokalajit, jotka vaativat N puuttuvaa tai osittain puuttuvaa ainetta. Reseptejä voidaan hakea myös niiden sisältämien ruoka-aineiden perusteella.

Aineisiin voidaan liittää merkintä niiden sisältämistä allergeeneistä, jotta voidaan rajoittaa hakua, jos halutaan välttää esimerkiksi maitoa sisältäviä ruokalajeja. Aineet voidaan myös rakentaa itse reseptistä samoin kuin ruokalajit, esimerkiksi jauhelihapihvin resepti voi sisältää lihamureketaikinaa, jolla on puolestaan oma reseptinsä. Jos lihamureketaikina puuttuu, ohjelma yrittää koota sen raaka-aineistaan. Ohjelma käsittelee sekä reseptejä että ruoka-aineita ”aineina”: reseptit ovat aineita, joita voidaan valmistaa muista aineista, joita kutsutaan ohjelmassa ainesosiksi. Ainesosat, joilla ei ole omia ainesosiaan, kutsutaan raaka-aineiksi.

Ohjelmassa on myös luokka, Muuntaja, joka tekee yksikkömuunnokset aineille niiden tiheyttä käyttäen. Ohjelmassa kaikkien aineiden tiheyden mittayksikkö on g/ml. Muuntaja on olennainen, koska resepteissä aineet voivat ilmetä eri mittayksikössä kuin tavallisesti. Esimerkiksi jauhoja voidaan ostaa kiloittain, mutta reseptit voivat mitata niitä desilitroina.

Ohjelmassa on graafinen käyttöliittymä, jolla voi käyttää kaikkia ohjelman toimintoja, kuten hallita varastoa ja luoda uusia aineita. Ohjelma on siis pyritty toteuttamaan keskivaikealla tasolla.

Käyttöohje

Ohjelma käynnistetään GUI-objektin kautta. Siitä aukeaa ohjelman pääikkuna, jossa on tekstitaulukko, jossa on listattuna kaikki ohjelman tuntemat aineet sekä niiden määrät ja allergeenit. Ikkunassa on myös kuusi nappia: Reseptihaku, Luo Resepti, Varastonhallinta, Avaa aine, Tiheyslaskuri ja Sulje Ohjelma. Nämä avaavat ikkunoita, joissa voidaan suorittaa erinäisiä toimintoja. Reseptihaussa voidaan hakea eri aineita, reseptinluonnissa voidaan luoda uusia aineita ja varaston hallinnassa voidaan muuttaa aineiden määriä ohjelman muistissa. Avaa aine -napilla pääsee tarkastelemaan aineiden ominaisuuksia ja muuttamaan niitä. Tiheyslaskuri helpottaa aineiden luontia laskemalla annetun massan ja tilavuuden avulla niitä vastaavan

tiheyden. Sulje ohjelma tallentaa ohjelman tiedot ja sulkee ohjelman.

Aine-olioiden kaikkia muuttujia pystytään muokkaamaan eri metodeilla käyttöliittymän Aine-näkymän kautta. Ikkunasta näkee aineen tiedot ja voi muokata sen ainesosia, allergeeneja ja muita ominaisuuksia sekä selvittää mistä raaka-aineista se koostuu. Ikkunassa on napit näitä toimintoja varten, ja niitä painettua ohjelma pyytää käyttäjältä merkkijonona annetun komennon, jonka se pyrkii toteuttamaan. Komennot ovat tyypillisesti muotoa [operaattori] [ominaisuus] [uusi arvo]. Käyttö on pyritty selittämään kussakin pyynnössä mahdollisimman selkeästi.

Varaston Hallinta -nappi avaa pienen ikkunan, jossa on neljä nappia varaston hallintaan. Nämä ovat aineen määrän ja mittayksikön muuttamiseen, mutta myös yksittäisten tai kaikkien aineiden poistamiseen. Reseptinluonnissa käyttäjä voi luoda uusia aineita täyttämällä ikkunan tekstikentät, jotka ovat: Aineen nimi, allergeenit, tiheys, määrä ja mittayksikkö, sekä Aineen kuvaus. Näistä ei ole pakko täyttää muita kuin nimi. Nimen voi antaa missä muodossa tahansa, mutta ohjelma muuttaa sen yhtenäiseksi ja pikkukirjaimiseksi merkkijonoksi. Allergeenit erotetaan pilkulla. Tiheys, määrä ja mittayksikkö erotetaan pilkulla, eli annetaan muodossa Double, Double, String. Tiheyden ja määrän tulee olla vähintään 0.0 ja jos ohjelma ei tunnista annettua mittayksikköä, se muutetaan ”kpl”-muotoon. Mittayksikön voi muuttaa aineikkunasta. Viimeiseen kenttään voi kirjoittaa aineen kuvauksen.

Ohjelma tunnistaa vain yleisimmät massan ja tilavuuden yksiköt, sekä määritelmän ”kpl” joka kuvaa kappaleina tai annoksina mitattavia aineita. Ohjelman käyttämä tiheyden yksikkö on g/ml, jota se käyttää tehdessään yksikkömuunnoksia. Kappalemittaisille aineille ei voi tehdä yksikkömuunnoksia. Tunnetut mittayksiköt ovat:

- gramma ("g")
- kilogramma ("kg") = 1000.0 g
- naula ("lb") = 453.6 g
- unssi ("oz") = 28.4 g
- millilitra ("ml")
- desilitra ("dl") = 100.0 ml
- litra ("l") = 1000.0 ml
- teelusikka ("tl") = 5.0 ml
- ruokalusikka ("rkl") = 15.0 ml
- kuppi ("cup") = 240 ml
- pintti ("pint") = 470 ml

Tiheys, erityisesti muodossa g/ml, on melko hankala käsittää intuitiivisesti, kun on luomassa aineita. Sen vuoksi ohjelmaan on lisätty tiheyslaskuri. Tiheyslaskuri pyytää käyttäjältä syötteenä massan arvon ja

mittayksikon sekä tilavuuden arvon ja mittayksikön, ja laskee näiden suhteen avulla käyttäjälle tiheyden arvon.

Ohjelman rakenne

Ohjelma on jaettu kahdeksaan pääosaan: Aine-luokkaan, Hakukone-, Muuntaja-, Varasto-, IO-, UI- ja GUI-yksittäisolioihin sekä Poikkeukset poikkeusluokkiin. Näistä keskeisimmät ovat Aine, Varasto ja Hakukone; muut luokat lähinnä helpottavat näiden toimintaa.

Aine-luokka mallintaa reseptejä, ainesosia ja raaka-aineita, eli käytännössä kaikenlaisia aine-olioita. Aine-olioihin on tallennettu kaikki aineille ominainen tieto: nimi, allergeenit, tiheys, määrä (joka syntyy valmistettaessa), oletusmittayksikkö sekä ainesosat, eli mitä aineksia vaaditaan aineen valmistukseen. Lopuksi on myös aineen kuvaus, joka on yleisimmiten valmistusohje aineelle.

Varasto-olioon tallennetaan tiedot kaikista ohjelman tuntemista Aine-olioista, ja tieto kuinka paljon käyttäjällä on niitä varastossa. Varastolla on metodeja, joilla voidaan hallinoida näitä määriä ja välittää näitä tietoja eteenpäin erityisesti Hakukoneelle.

Hakukone-olio on työkalu, joka helpottaa reseptien löytämistä Varaston tiedoista. Sen metodeja käytetään palauttamaan ainelistoja. Aineita voidaan suodattaa nimen ja allergeenien perusteella. Kriteeriksi voidaan myös määritellä, että aine pitää pystyä valmistamaan varastossa olevista aineksista, tai siten että puuttuu n ainesta.

Muuntaja-oliota käytetään tekemään muunnoksia eri massojen ja tilavuuksien mittayksiköiden välillä. Hakukone käyttää Muuntajaa tarkistaessaan riittävätkö Varaston ainekset, koska Varastossa olevat aineet voivat olla eri mittayksikössä kuin reseptissä jota Hakukone tarkistaa.

IO-olio käsittelee ohjelman tietojen tallennustarpeita. Se tallentaa Aine-olioita omille tekstitiedostoilleen ja Varaston tiedot yhdelle tekstitiedostolle. Se voi myös välittää tiedot Varastolle tältä tiedostolta, ja luoda Aine-tiedostoilta uusia Aine-olioita. IO suorittaa näitä toimenpiteitä, kun ohjelma käynnistetään tai suljetaan, tai kun luodaan uusia aineita käyttöliittymän kautta.

GUI-olio on ohjelman graafinen käyttöliittymä, joka kutsuu pitkälti UI-olion metodeja. Sen kautta voidaan käyttää kaikkia ohjelman toimintoja, ja huomattavasti kätevämmin kuin merkkipohjaisesti, joskin jotkut toiminnot vaativat tekstipohjaisia komentoja. Käyttöliittymässä on viisi suurempaa näkymää, muut osat ovat lähinnä ponnahdusikkunoita ynnä muita vastaavia. Pääikkunassa on listattuna taulukkoon kaikki Varaston tuntemat aineet ja niiden määrät, sekä napit joista pääsee kolmeen muuhun ikkunaan. Reseptin luonti-

ikkunassa voidaan interaktiivisesti luoda uusia Aineolioita, ja muuttaa olemassa olevia. Reseptihaussa voidaan käyttää Hakukonetta aineiden etsimiseen. Varastonhallinnassa voidaan kätevästi muuttaa aineiden määriä Varastossa. Viimeinen ikkuna on Aineikkuna, jonka voi avata painamalla listattuja aineita eri näkymissä. Siinä voi tarkastella ja halutessa muuttaa aineen ominaisuuksia.

Algoritmit

Valtaosa Muuntaja-olion metodeista toimii käyttäen kaavaa tiheys = massa / tilavuus (d = m / V). Tätä kaavaa hyödyntääkseen Muuntaja laskee eri mittayksiköiden suhteet (esim. kg/g = 1000), josta voi laskea muunnoksia, kun on kyse muunnos massayksiköstä massayksikköön tai vastaava tilavuuksien välillä. Tilavuuksien ja massojen välisiä muunnoksia varten yksiköt muutetaan vastaamaan Aine-olioiden käyttämää tiheyden yksikköä (g/ml).

Esimerkiksi, jos halutaan selvittää mikä aineen määrän arvo on toisessa mittayksikössä, kutsutaan Muuntajan muunna-metodia. Se ottaa aineelta sen tiheyden ja mittayksikön, ja välittää ne laske-metodille. Laske-metodi kutsuu metodeja, jolla tunnistetaan minkälainen muunnos on kyseessä, jonka jälkeen kutsutaan oikeaa muunnosmetodia. Nämä metodit laskevat aineen tiheyden avulla uuden massan tai tilavuuden arvon, ensiksi perusmittayksikössä g/ml, jota verrataan kohdemittayksiköön. Saadun suhteen avulla lasketaan lopputulos.

Aine-luokan aineetYhteensa -metodi, jolla selvitetään reseptin perusraaka-aineet ja määrät, käyttää rekursiota. Se tarkistaa ensin ovatko sen ainesosat raaka-aineita, ja ne jotka ovat lisätään tuloskokoelmaan. Jos ainesosa ei ole raaka-aine metodia kutsutaan sille rekursiivisesti, ja niin edelleen. Löydetyt raaka-aineet lisätään aina edelliseltä aineelta löydettyyn raaka-ainekokoelmaan, kunnes jokainen aines on käyty läpi.

Hakukoneen hae -metodi käy läpi varaston aineet. Se lisää aineen listaan, jos sitä on valmiina varastossa, jos sen voi valmistaa aineksistaan, jos se on raaka-aine (ja käyttäjä sallii puuttuvat aineet), tai jos aineen raaka-aineista puuttuu korkeintaan n kappaletta. Näistä hakutuloksista voidaan halutessa suodattaa pois aineita allergeenien ja nimen perusteella.

Metodi voiValmistaa tarkistaa käy läpi aineen ainesosat ja selvittää ensin onko sitä tarpeeksi valmiina valmistusta varten, ja jos ei, se tarkistaa rekursiivisesti voiko tätä puolestaan valmistaa omista aineksistaan. Lopussa tarkistetaan riittävätkö ainekset.

Tietorakenteet

Varasto käyttää hakurakennetta (Map), koska se koostuu käytännössä täysin avain-arvo -pareista (ainemäärä).

Muut tietorakenteet ovat lähinnä muuttuvatilaisia kokoelmia, koska halusin toteuttaa aineet siten, että niiden ominaisuuksia voi halutessa muuttaa tai korjata. Jälkikäteen ajatellen monet näistäkin olisivat voineet olla hakurakenteita.

Aine-luokka sisältää monenlaista tietoa, jotka mallintavat erilaisia reseptejä ja ruoka-aineita. Nämä ovat pääasiallisesti Scalassa käytettäviä tavallisia teksti, numero ja kokoelmaolioita. Ainoa hieman erilailla toimiva muuttuja on aineen mittayksikkö: se on String-muodossa, mutta yleensä kun sitä käytetään, käytetään Muuntaja-luokkaa tunnistamaan mikä yksikkö on kyseeessä ja mitä numeroarvoa se vastaa kussakin tilanteessa. Sille olisi ehkä voinut toteuttaa jonkinlaisen case class -rakenteen, mutta päädyin siihen että String ajaa saman asian.

Tiedostot

Ohjelma käsittelee IO-olion kautta kahdenlaisia tekstitiedostoja: Aine- ja Varasto-tiedostoja. Aine-oliot tallennetaan tekstitiedostoille, Reseptikirjan reseptikansioon. Sieltä ne voidaan lukea myöhemmin tarvittaessa. Tiedoston ensimmäiselle riville tulee aineen nimi. Toiselle riville tulee aineen tiheys, määrä ja mittayksikkö. Seuraaville riveille tulevat ainesosat, jokainen omalle rivilleen. Ainesosien jälkeen seuraavalla rivillä on tähtimerkki ("*") erottimena, jonka jälkeen seuraavalla rivillä on aineen allergeenit. Viimeisillä riveillä on aineen kuvaus. Esimerkki:

Spagetti bolognese
0.0,4.0,kpl
spagetti,300.0,g
kastike,800.0,g
*
liha,tomaatti
Spagetti bolognese, neljä annosta. Paista jauheliha... jne.

Varaston tiedot tallennetaan riveittäin muodossa nimi ja määrä. Esimerkki:
spagetti,300.0
maito,1.0
kananmuna,12.0

Testaus

Ohjelman testaus oli mielestäni riittävää, mutta yksikkötestauspuoli oli rajallista. Alun perin viime vuoden kurssilla minulla oli paljon ongelmia saada scalatest-kirjasto toimimaan, ja yksikkötestit jäivät vähäisiksi. Toteutin niitä nyt jonkin verran lisää, mutta ohjelmassa on hyvin paljon toimintoja, joita varten ei ole laadittu yksikkötestausta. Loppujen lopuksi päädyin tarkistamaan ohjelman ajamalla sen kaikki toiminnot läpi itse, seuraten, että kaikki toimii.

Suoritin kaikki suunnittelemani testit, mutta en tehnyt kaikille niistä yksikkötestejä. Tein yksikkötestejä lähinnä metodeille, jotka muuttavat tietoa ja muistia ohjelmassa. Suunnitellut testit eivät olleet riittäviä, sillä huomasin keksiväni monenlaisia erikoistilanteita ohjelman ajossa, joita en ollut osannut ajatella etukäteen. Ohjelma läpäisi testaukseni – seuraavassa osassa mainittuja puutteita lukuunottamatta – joten

en usko, että testauksessani olisi kovin suuria aukkoja.

Ohjelman tunnetut puutteet ja viat

Hakukone-olio on ohjelman puutteellisin osa. Erityisesti metodi voiValmistaa, jolla tarkistetaan voiko annetun aineen valmistaa varastossa olevilla aineilla, on vajavainen. Tällä hetkellä se palauttaa oikean arvon, kun kyseessä on aine, jolla ei ole ainesosia, tai sen ainesosat ovat raaka-aineita (eli ainesosilla ei ole omia ainesosia). Väärä tulos saattaa tulla, kun esimerkiksi reseptiin A tarvitaan 2 ainetta B, jonka valmistamiseen puolestaan tarvitaan 4x C. Tämä virhe syntyy, koska metodi ei tee tarkistusta paljonko ainetta B syntyy kun sitä valmistetaan aineesta C. Hakukone ei siis anna välttämättä oikeita tuloksia, kun yritetään rajata, kuinka monta ainesosaa saa puuttua reseptistä. En keksinyt ratkaisua tähän ongelmaan.

Ajon aikaisia virheitä pystyy luultavasti aiheuttamaan, jos rikkoo Aine-olioiden tiedostoja, esimerkiksi lisäämällä aineelle ainesosia, ja poistamalla ne kesken ajon. Ohjelma poistaa ainesosaluettelosta puuttuvat ainesosat yleensä vain ohjelmaa käynnistettäessä. Ohjelma pystynee käsittelemään varasto-tiedoston korruptoimisen ajon aikana, mutta Aine-tiedostot ovat melko riippuvaisia toisistaan.

Ohjelmassa ei ole ehkä tarpeeksi lisämäärittelyitä, kuten private, varsinkin koska suurinta osaa muuttujista voi joka tapauksessa muuttaa jollakin saman luokan metodilla.

Kolme parasta ja kolme heikointa kohtaa

Mielestäni ensimmäinen vahva kohta ohjelmasta on tekstitiedostot ja niiden käsittely IO-objektissa. Aine-tiedostoja pystyy kirjoittamaan nopeasti tekstitiedostolle, joskin muuttujien järjestys pitää tietää muuta kautta, koska niitä ei ole nimetty. Varasto-tiedosto on myös hyvin suoraviivainen. Tämä oli hyödyllistä testatessa ohjelmaa ilman käyttöliittymää, sillä se oli huomattavasti nopeampaa kuin kirjoittaa konstruktoreita yhteen pötköön Aine-olioille. Olen tyytyväinen IO-objektiin, koska se pystyy käsittelemään näitä tiedostoja, vaikka ne olisivat korruptoituneita. Luultavasti olisi parempia ja tehokkaampiakin toteutuksia kuin omani, mutta nämä tuntuvat toimivan hyvin.

Toinen vahva kohta on mielestäni koodin luettavuus. Se ei ole täydellistä, ja jotkut kohdat ohjelmasta ovat sekavempia kuin toiset, mutta ymmärsin miten kaikki toimii, kun palasin jälleen ohjelman pariin lukemaan vanhaa koodiani. Mielestäni ohjelma on suurimmaksi osaksi selkeä lukea ja ymmärtää mitä eri metodeissa tapahtuu. En kommentoinut kaikkea täysin tyhjentävästi, mutta mielestäni nimesin metodit ja niiden konstruktorit selkeästi.

Kolmas vahva kohta on poikkeusten käsittely. Tein muutaman oman poikkeusluokan, ja niitä olisi kenties voinut olla enemmän, mutta olen tyytyväinen, että ne tuntuvat nappaavan kaikki eri virheet, joita tuli vastaan ohjelman testauksessa. Käyttäjälle tulee näistä myös graafisen käyttöliittymän kautta palaute, joten

suurimmassa osassa tilanteita pitäisi olla melko selkeää, mikä meni pieleen.

Ensimmäinen heikkous on Hakukone. Se täyttää suurimman osan toiminnallisuudesta, mutta niin kuin mainittu viat ja puutteet -kohdassa, siinä on tapauksia, jossa se ei toimi täysin oikein. Puutteet eivät välttämättä ilmaannu arkisemmissa ruokalajeissa, mutta jos lähdetään lisäämään moniasteisia reseptejä, virheellisten hakutulosten riski kasvaa. Se oli vaikein osa ohjelman toteutuksessa ja harmillisesti se jäi vajavaiseksi.

Toinen heikkous on että käyttöliittymä on monelta osaa merkkipohjainen, vaatien paljon merkkijonona annettavia syötteitä, joiden tulee olla melko tarkkaan oikein annettuja, muuten ne eivät toimi. Kohdissa, joissa syötettä vaaditaan, on pyritty selittämään käyttö, mutta olisin mieluummin toteuttanut interaktiivisemman käyttöliittymän, mutta valitettavasti en saanut sellaista toimimaan Swing-osaamisellani.

Kolmas heikkous on monien algoritmien tehottomuus. Erityisesti Hakukoneen metodeissa käydään kokoelmien kaikki alkiot läpi. Käytännössä testauksessani ohjelman muistin käyttö oli pientä, joten se ei muodostunut ongelmaksi, mutta periaatteessa ohjelma voisi hidastua hyvin paljon, jos siihen lisättäisiin paljon monimutkaisia reseptejä.

Poikkeamat suunnitelmasta ja aikataulu

Käytin vuosi sitten tekemääni projektia, joten jätän tähän viime vuoden dokumenttini aikataulupohdinnan perään, kuten teknisessä suunnitelmassanikin. Tänä vuonna aikomukseni oli parannella erityisesti Hakukonetta ja GUI:ta, joiden uudelleen suunnitteluun käytin yhteensä noin 8 tuntia, hyvin pienellä tuloksella. Yritin tutkia Swing-kirjastoa, mutta en valitettavasti saanut haluamiani taulukoita toimimaan, ja ohjelmaan jäivät karut tekstimuotoiset taulukot. En myöskään onnistunut ratkaisemaan Hakukoneen metodien puutteita.

Loppujen lopuksi käytin suurimman osan ajasta viime vuoden arvostelussa mainittujen kehityskohtien korjaamiseen, kenties noin 15 tuntia. Sain ne mielestäni suurimmaksi osaksi korjattua, mutta jotkin asiat, kuten yksikkötestaus, jäivät jossain määrin puutteellisiksi.

Ajankäyttöarvioni osui melko tarkasti kohdilleen, joskin vikojen korjauksessa tuntui, että niitä ilmeni koko ajan lisää, joten minun olisi pitänyt aloittaa ne aikaisemmin, jotta olisi jäänyt enemmän aikaa uusien ominaisuuksien toteutukseen.

Vanha:
Alun perin suunnittelin, että ainesosat-muuttuja on Aine-luokan konstruktori, mutta siirsin sen tavalliseksi

muuttujaksi. Tämä oli sen vuoksi, että Aine-olioiden luominen on huomattavasti hankalampaa, jos ohjelman täytyy tietää aineen kaikki ainesosat ennen luomista. On helpompi luoda aine ensin, ja luoda ainesosat halutessa, ja sitten lisätä ne reseptiin.
Päädyin assistentin neuvosta myös jakamaan IO-yksittäisolion erilleen, mikä oli hyvä idea. Alunperin en ajatellut toteuttavani UI-oliota, mutta se olisi ollut virhe, ja tarpeettomasti monimutkaistanut GUI-olion koodia.
Ajankäyttösuunnitelmani ei osunut ollenkaan kohdilleen. Minulla meni Aine-, Varasto- ja Hakukone-luokissa luultavasti noin puolet siitä ajasta, jonka arvioin. Sen sijaan Muuntaja-luokassa kesti muutama tunti pitempään kuin suunnittelin. Graafisessa käyttöliittymässä meni merkittävästi kauemmin kuin odotin. Yritin toteuttaa liian monimutkaisia ominaisuuksia Swingillä aikatauluuni ja taitoihini nähden. Minulla tuli eteen ongelmia, jotka pysättivät täysin työskentelyni etenkin käyttöliittymän kanssa. Sain toteutettua muut luokat sen verran hyvässä tahdissa, etten varannut tarpeeksi aikaa käyttöliittymää varten.
Lähdin työstämään projektia liian myöhään, joka kostautui, kun eteen ilmestyi odottamattomia haasteita, joista suurin osa oli itse aiheutettuja. Työjärjestykseni oli suurin piirtein sama kuin suunnittelin, poikkeuksena Muuntaja, jonka toteutin ennen kuin viimeistelin Aine-luokan.
Aikataulutu oli siis erittäin huono, ja venyi pahasti yliajalle.

Arvio lopputuloksesta

Olen melko tyytyväinen ohjelmaan. Se on hyvin ruma ulkoisesti, ja hieman epäintuitiivinen käyttää, mutta se toimii. Onnistuin korjaamaan suurimman osan ongelmista, jotka vaivasivat sitä viime vuoden palautuksessani. Oleellisin ohjelmassa jäljellä oleva puute on Hakukone: siinä on erikoistilanteita, joissa se ei anna täysin oikeita tuloksia. Muuten ohjelmassa on kattavasti metodeja reseptien ja varaston käsittelyyn.

Ohjelman rakenne on melko modulaarinen ja sitä olisi jatkossakin luultavasti melko helppo laajentaa ja kehittää. Esimerkiksi käyttöliittymää pystyisi parantamaan melko paljon ilman, että muita ohjelman osia tarvitsisi muuttaa. Olen tyytyväinen ohjelman luokkarakenteeseen ja ratkaisumenetelmiin, mutta tietorakenteiden puolesta epäilen, että olisi parempia ratkaisuvaihtoehtoja. Esimerkiksi jotkut Aine-olioden kokoelmat olisivat voineet olla Map-muotoisia, joka olisi tehokkaampi kuin käydä läpi pitkiä listoja.

Jos aloittaisin projektin alusta, en muuttaisi yleisrakenteesta kovin paljoa. Toteuttaisin yksikkötestausta hyvin paljon heti alusta lähtien, koska olisin säästänyt monen metodin kohdalla paljon aikaa, jos olisin huomannut heti testaamalla, että ne toimivat täysin oikein. Miettisiin tarkemmin parhaan vaihtoehdon kokoelmien suhteen ohjelman eri osissa.

Viitteet

Ohjelmointi 1 -kurssimateriaali
Ohjelmointistudio 2 -kurssimateriaali

Ohjelmointi 2 -kurssimateriaali
scala API
scala.swing API
stackexchange.com
tutorialspoint.com/scala

Liitteet

Smart_Cookbook -Scala-projekti